

AD-A052 741 SRI INTERNATIONAL MENLO PARK CA COMPUTER SCIENCE LAB F/G 9/2
THE RELATIONSHIP OF HDM TO THE NAVY'S SOFTWARE-DEVELOPMENT PROC--ETC(U)
MAR 78 L ROBINSON N00123-76-C-0195

UNCLASSIFIED

SRI/CSL-70

NL

1 OF 1
AD
A052741



END
DATE
FILMED
5-78
DDC

ADA 052741

DDC FILE COPY

THE RELATIONSHIP OF HDM TO THE NAVY'S SOFTWARE- DEVELOPMENT PROCESS

Technical Report CSL-70
Deliverables A009, A014, A015

March 1978

12
APR

By: Lawrence Robinson

Prepared for:

Naval Ocean Systems Center
San Diego, California 92152

Contract N00123-76-C-0195

SRI Project 4828



SRI International
333 Ravenswood Avenue
Menlo Park, California 94025
(415) 326-6200
Cable: SRI INTL MPK
TWX: 910-373-1246



This document has been approved
for public release and sale; its
dissemination is unlimited.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
		12 13p.
4. TITLE (and Subtitle)		5. TYPE OF REPORT & PERIOD COVERED
THE RELATIONSHIP OF HDM TO THE NAVY'S SOFTWARE-DEVELOPMENT PROCESS.		9 Technical Report.
7. AUTHOR(s)		14 15 SRI CSL-70 SRI Project 4828
10 Lawrence Robinson		8. CONTRACT OR GRANT NUMBER(s)
11 SRI International ✓ Computer Science Lab. 333 Ravenswood Avenue Menlo Park, California 94025		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Deliverables A009, A014, A015
12. CONTROLLING OFFICE NAME AND ADDRESS Naval Ocean Systems Center San Diego, California 92152		11 12. REPORT DATE Mar 78 13. NO. OF PAGES 13
14. MONITORING AGENCY NAME & ADDRESS (if diff. from Controlling Office)		14. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE n/a
16. DISTRIBUTION STATEMENT (of this report) Approved for public release; distribution unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from report) n/a		
18. SUPPLEMENTARY NOTES A038-51 F		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) embedded systems, Hierarchical Development Methodology (HDM), procurement, Program Design Specification (PDS), Program Performance Specification (PPS), software-development process, software-development tools, system specifications, Statement of Work (SOW)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) HDM (Hierarchical Development Methodology) is a set of concepts, languages, and on-line tools that is intended to aid in the production of reliable, correct, and maintainable software. This document describes the relationship between HDM and the Navy's software-development process, to illustrate that HDM is consistent with — and can even enhance — the Navy's software-development process.		
410 646 —		APR 17 1978

CONTENTS

ACKNOWLEDGMENTS	2
I INTRODUCTION	3
II A BRIEF DESCRIPTION OF HDM	4
III THE RELATIONSHIP OF HDM AND THE PPS	6
IV THE RELATIONSHIP OF HDM AND THE PDS	7
V THE RELATIONSHIP OF HDM AND THE SOW	9
VI CONCLUSIONS	10
REFERENCES	11

ACCESSION for			
NTIS	White Section <input checked="" type="checkbox"/>		
DDC	Buff Section <input type="checkbox"/>		
UNANNOUCED <input type="checkbox"/>			
JULY 1964 1971			
BY			
DISTRIBUTION/AVAILABILITY CODES			
Dist.	Avail.	and/or	SP CIAL
<i>P</i>			

ACKNOWLEDGMENTS

The author wishes to thank Jack Goldberg and Karl Levitt of SRI, and Lin Sutton of NOSC, for their critical reading of this report.

I INTRODUCTION

HDM (Hierarchical Development Methodology) was created at SRI International, with support from the Navy under Contract N00123-76-C-0195 (Methodology for Modular Operating Systems), to aid in the development of embedded computer systems that are more correct, reliable, and maintainable than those produced by conventional methods. In an unrestricted environment, HDM has the potential of greatly aiding software development. However, the Navy has particular needs associated with its own software-development process [1] *. Thus, for HDM to be of benefit to the development of Navy software, it must be possible to integrate HDM into the Navy's software-development process without sacrificing either the special merits of HDM or the integrity of Navy procedure.

The goal of this report is to show that the use of HDM is consistent with the Navy software-development process, and how some of the information provided by HDM (e.g., formal specifications), can be used directly in Navy software documents to supplement or replace the information normally found there. The conclusions are that HDM and the standard Navy documents can be integrated with no trouble, resulting in more standardization of the documents than would be possible using the current guidelines alone.

This report first briefly describes HDM, and then characterizes a possible relationship between HDM and the standard Navy systems documents known as the PPS (Program Performance Specification), the PDS (Program Design Specification), and the SOW (Statement of Work).

* References are listed at the end of the report.

II A BRIEF DESCRIPTION OF HDM

HDM is an integrated set of concepts, languages, and tools designed to enable the production of large software systems that are correct, reliable, and maintainable. It is based on the following concepts:

- * Structuring a system as a hierarchy of independent units, whose interconnections to other units of the hierarchy are well defined. The structuring involves data as well as control.
- * Formally specifying -- in a precise, mathematical language -- each of the components of a system by stating what component's function, independently of how the component performs the specified function.
- * Formally verifying, via a mathematical proof, that the implementation of a given system meets its specifications.

These principles result in better designs, more reliable systems, and easier maintenance.

HDM divides the software-development process into the following stages:

- * Conceptualization, in which the problem to be solved is formulated in detail.
- * External Interface Definition, in which the external interfaces of the system, their component parts, and the functions of each part, are identified.
- * Intermediate Interface Definition, in which the parts of the internal interfaces of the system, and the component functions of each part, are identified.
- * Formal Specification, in which formal specifications are written for each part of the system.
- * Formal Representation, in which the data structures of a component of the hierarchy are defined in terms of the data structures of more primitive components.
- * Abstract Implementation, in which abstract (descriptive) programs are written to implement each function of the system.
- * Coding, in which the abstract programs are translated into runnable source code.

- Formal Verification, in which the consistency of the specifications and the abstract programs is formally proved. This stage is optional, and is performed only in cases where the reliability of the system is important enough to justify its large cost.

More information on HDM can be found in [2] and [3].

III THE RELATIONSHIP OF HDM AND THE PPS

The PPS (Program Performance Specification) is a statement of the requirements -- performance, functional, and equipment -- of an embedded system. The PPS is generated before the software is designed, and should not contain any discussion of design or implementation issues.

The PPS properly corresponds to the first (conceptualization) stage of HDM. HDM currently provides no language for formally stating requirements, so -- in general -- there is no document of HDM that can be used in the PPS. However, if the use of a particular piece of hardware (e.g., processor, disk drive, terminal) is specified as part of the requirements, its formal specifications can be written using HDM and inserted into the PPS. In addition, HDM provides a particular way of looking at requirements and often provides criteria for stating them more cleanly (for example, separating them from design issues).

Thus, HDM provides a conceptual basis for the PPS and formal specifications for any hardware components that are part of the system requirements.

IV THE RELATIONSHIP OF HDM AND THE PDS

The PDS (Program Design Specification) is a statement of the design details of an embedded system. It always contains a description of the structure of the system, and in many cases uses flowcharts to specify the functions of system components. It is assumed that a team of programmers, using only the PDS, could be assigned to code, integrate, and maintain a software system.

The PDS corresponds to the HDM stages of external interface definition, internal interface definition, formal specification, formal representation, and abstract implementation. The correspondence between the PDS and the first three of these stages is clear. The outputs of these stages describe the structure of the system and the behavior of each component, and should be definitely included in the PDS as requirements to be met by the implementations of the system. However, the correspondence between the PDS and the last two of the above stages is open to interpretation. The outputs of these stages describe the definitions of data structures of various components of the hierarchy and the implementations of all functions in the system, and should also be included in the PDS. However, should they be interpreted as strict requirements or merely guidelines? The answer will probably vary according to individual circumstances.

One advantage of incorporating HDM specifications into the PDS is that HDM specifications are more uniform and more precise than those of conventional techniques, including flowcharts and pseudocode. Thus, using HDM, the PDS becomes more precise and uniform. The incorporation of documents produced using HDM enhances the major goal of the PDS, i.e., to communicate the intent of the system designers to those who must implement, integrate, test, and maintain the system.

Thus, the bulk of the outputs of HDM should be included in the PDS, since they form a basis for implementation and maintenance of the system. The required sections of the PDS concerning the approaches to programming, integrating, and maintaining the system are directly derivable from the rules of HDM. As a result, there is little in the PDS that HDM does not provide.

V THE RELATIONSHIP OF HDM AND THE SOW

The SOW (Statement of Work) is an integral part of the process by which the Navy procures embedded systems from outside sources. It can specify both the tasks to be performed and the method for performing them.

HDM can assist in the procurement of software systems in several ways:

- * In the procurement of an entire system, including both design and implementation, the use of HDM can be required. The use of HDM will aid in monitoring the progress of the contract, because of the formal specifications and tools that aid in the design process. In addition, the system produced will probably be better, and certainly more easy to maintain.
- * In the procurement of the implementation for a previously designed system, HDM specifications can be generated as the output of the design process and can be used in the RFQ as requirements to be placed on the implementation effort. Such formal specifications provide more information, in a more precise way, than do conventional English specifications.
- * In the procurement of the verification of a system, HDM can be required as the method for specifying and formally proving the desired properties. There are two types of verification supported by HDM: design verification, in which certain properties (e.g., security, reliability) are proved based only on the specifications for the design of the system; and implementation verifications, in which the system's implementation is proved consistent with the specification. HDM can be used in both types of verification.

Thus, the formal specifications and precise thinking required by HDM can definitely aid the Navy in procuring satisfactory software systems.

VI CONCLUSIONS

Although HDM provides a new framework for thinking about the structure of embedded systems, it divides the software-development process into stages that are entirely consistent with the stages of the Navy software-development process. In fact, software development using HDM actually enhances the Navy software-development process, because the outputs of HDM fit readily into Navy documents.

REFERENCES

1. Program Management Manual, NELCINST 5000.2, Naval Electronics Laboratory Center, San Diego, California (January 1976).
2. L. Robinson, "Hierarchical Development Methodology -- Command and Staff Manual," Technical Report CSL-49, Contract N00123-76-C-0195, SRI Project 4828, SRI International, Menlo Park, California (March 1978).
3. K. N. Levitt and L. Robinson, "Handbook for Hierarchical Development Methodology," Technical Report CSL-72, Contract N00123-76-C-0195, SRI Project 4828, SRI International, Menlo Park, California (March 1978).

DISTRIBUTION LIST

Defense Documentation Center Cameron Station Alexandria VA 22314	12 copies
Mr. Tony Allos Code 6201 Naval Ocean Systems Center 271 Catalina Boulevard San Diego, CA 92152 (with cover letter through Contracts)	1 copy
Mr. Lin Sutton Code 8223 Naval Ocean Systems Center 271 Catalina Boulevard San Diego, CA 92152	35 copies
Mr. William L. Carlson Defense Advanced Research Projects Agency Information Processing Technology Office 1400 Wilson Boulevard Arlington, VA 22209	15 copies
Mr. Neal Hampton Code 8223 Naval Ocean Systems Center 271 Catalina Boulevard San Diego, CA 92152	15 copies
Ms. Marlene Hazle MITRE Corp. Box 208 Bedford, MA 01730	1 copy
Professor Stuart Madnick MIT Sloan School of Business E53-330 Cambridge, MA 02139	1 copy
Mr. John Machado Naval Electronic Systems Command National Center No. 1 Crystal City Washington, DC 20360	5 copies
Mr. William Rzepka ISIM Rome Air Development Center Griffiss Air Force Base	5 copies

Rome, NY 13441

Dr. Harold S. Schwenk, Jr.
BGS Systems, Inc.
P.O. Box 128
Lincoln, MA 01773

1 copy